

# State-of-the-Art Report in Web-based Visualization

EuroVis 2016 STAR 论文类型：综述

---

## 动机

- 基于网页的可视化应用利于跨平台使用
- 非可视化专业用户可免去安装烦恼，即开即用

## 文章概览

- Web平台的可视化实现技术梳理
- 最新的实现技术（WebGL等）
- 相关工作分类

## 可视化应用的实现技术

- 可视化应用也是一种软件，需要遵从当前最新的技术现状，和发展规律。
- 通常Web应用都是异构架构，即不同Web服务的前后端编程语言、运行平台等均有差别。
- 图1说明了在不同网络带宽和客户端计算资源条件下，远程渲染（可视化）界面需要采用的技术类型分类。从基于模型参数到基于图像，中间逐渐过渡。

## VaaS（Visualization as a Service）

- Web service使得不同应用模块之间使用web协议进行数据交换，方便组合开发。
- 对于Visualization as a service，目前有以下三种架构类型，基于可视化流水线所实现的位置分类：
  1. 整条流水线全部放在服务器端，用户不可见。整个可视化应用作为单一功能存在。
    - 优点：效率高，对终端用户方便
    - 缺点：功能单一
    - 实例：Tableau Online，Spotfire Cloud
  2. 流水线拆分为最小单位的模块，每个模块单独作为一个service，模块可拼接。
    - 优点：灵活性高，高级用户喜爱
    - 缺点：效率问题（数据在service间传输）
    - 实例：VisComposer
  3. 上述两种结合使用
    - 考验设计能力
- 实现方案：SOAP和RESTful

## Grid-based Visualization

内容略老可忽略。

## Cloud-based Visualization

- 云平台的GPU加速技术
  - 基于视频的串流: Nvidia GRID
  - GPU渲染
  - GPGPU

## 本地浏览器渲染

- 技术方案:
  - Canvas (2D或WebGL)
  - SVG
  - 数据压缩、传输及并行处理: WebSocket, Web worker

## 应用示例

粒子可视化、体可视化、三维地理可视化

## 信息可视化

3D应用、WebGL绘制二维视图等。

## 可视化论文的系统实现分类

原文中表1。

## 总结及挑战

- 公共数据源
  - 易访问
  - 用户通常是普通民众
  - 主要挑战: 简化知识发现过程; 提供LoD方式的可视化以应对不同层次用户的需求
- 可视化层的独立: 让数据处理service负责数据处理, 可视化service仅负责mapping和rendering。
- 对于海量数据, Query-base Visualization是一种有效的解决方案, 降低了可视化阶段的数据量。
- 目前的可视化工具很难无缝嵌入到其他分析工具中(例如Hadoop、Spark等), 基本都是独立平台。
- Hybrid visualization: 浏览器端与服务器端的workload可以无缝调整。
- 其他:
  - 浏览器端GPGPU加速
  - 浏览器端面向2D可视化的加速